

Working with Big Data

Vancouver Summer Program (VSP)

Joshua Catalano (CIDER)

Centre for Innovative Data in Economics Research (CIDER)

July 17, 2023


What is Big Data?

The origin of the term big data is frequently credited to computer scientist John R. Mashey in the early 1990s.

The advent of the term was made necessary in part by the proliferation of digital data collection.

Generally speaking, big data is well-characterized by "The Three Vs"¹

- High **Volume**
- High **Velocity**
- Large **Variety**

¹This list of characteristic traits is credited to data analyst Doug Laney. 

Volume, Velocity, and Variety

Volume refers to the amount of data. As expected, big data is characterized by larger amounts of data.

Velocity is the frequency data arrives and by extension, the frequency that data needs to be incorporated and acted on. Big data is characterized by rapid flows of data and quickly incorporating new data into action.

Variety refers to the structure of data and relatedly, the types of data included. Whereas traditional datasets tend to be tabular, big data also encompasses less structured data such as text, audio, images, video, sensor readings, etc.

Discussion – The Three Vs

While this is by no means the only way to characterize big data, it serves as a useful starting point. What are some real world examples of the following?

- High volume data
- High velocity data
- High variety (or unstructured) data

Drawbacks of Big Data

Computational Considerations

- Analyzing and processing big data can be costly in terms of money, computational resources, and time.
- At a certain point, even storing, transferring, and managing requests for the data can require industrial data solutions.

Methodological Considerations

- Assumptions underlying standard statistical methods might not be reasonable in the big data setting.
- Applying standard methods might require introducing structure to data

Just Beginning

In this class, students will learn the fundamentals of coding, data preparation, and data analysis.

No coding experience is assumed.

When relevant, we will discuss how big data relates to the material and will eventually have the opportunity to work with relatively large datasets.

Course Goals

By the end of this course, students will be able to

- Code in Python with an understanding of programming fundamentals
- Use GitHub to store and collaborate on code
- Load, clean, manipulate, and visualize data effectively
- Fit regression and classification models on data
- Perform select numerical computations
- Understand some of the math behind these techniques
- Learn about coding on their own more easily

What is a Programming Language?

Generally speaking, a programming language is a language that can be interpreted by a computer as a series of commands that it then executes.

- Programming languages tend to be text-based.
- Hundreds of programming languages – a small proportion of those see most of the use
- The more people use a programming language, the better that language becomes (especially true for open-source languages).

What is Open Source Software?

Open source software stands in contrast to proprietary software.

- Free (not paid)
- Licenses still exist and ensure people and organizations have an incentive to contribute
- Easier to collaborate
- Package management systems makes it easier to share your own packages and download packages others have made into your coding environment.

Python

Python is a general purpose coding language with many benefits:

- Relatively simple to use and learn
- Readable
- General purpose (can be used for data analysis, websites, web scraping, etc.)
- Very popular – lots of prebuilt tools
- Open-source!

The Python Trade-off

- Since Python tries to do everything well, it is rarely the best for any single task.
- Common Joke: Python is “the best language for nothing but the second best language for everything.”
- Good language for muliti-faceted projects and for first-time coders.

Jupyter Notebooks

Jupyter Notebook is an in-browser interactive development environment (IDE). It allows you to display code, markdown-style text, figures, and data in a notebook-like structure.

- Supports many coding languages
- Allows for easy code segmentation
- Great for integrating figures and code into lectures, homework assignments, projects, and reports.
- We will use a UBC service to run our notebooks on the cloud, but you can download Jupyter to your computer if you prefer.

GitHub

Code Developers use GitHub (among other Git-centric services) to share and work on projects together. In fact, we have a course GitHub where you can find all of the course materials.

- Code scripts and other files are stored in a repository in the cloud
- Changes to the repository are tracked and can be reversed (version control done through Git)
- Multiple collaborators can simultaneously work on code (Git resolves conflicts between changes)
- Repositories can be easily copied and merged back together.

Syllabus

All of the course materials, including the syllabus, are stored on the course's GitHub page.

Today's Lab

In today's lab, we will get all the necessary pieces together so we can start coding.

Steps

1. Create a free GitHub account
2. Download GitHub Desktop
3. Log into Jupyter Open using your Campus-wide Login (CWL)
4. Create and save a basic Jupyter notebook
5. Create your own repository for the course via GitHub desktop
6. Commit and push that repository to the cloud
7. Clone the course repository

Step 1 – Create GitHub Account

To create a GitHub account:

- Go to www.github.com
- Click the "Sign up" button.
- Go through all the sign up steps including confirming your email address
- Go back to GitHub and login – make sure you remember your login details!
- **Optional:** There is a way for students to get free GitHub pro accounts. Not required for this class, but you can view details here or by searching the "GitHub Student Developer Pack" on Google.

Step 2 – Download GitHub Desktop

To download GitHub desktop:

- Go to <https://desktop.github.com/> or Google GitHub Desktop
- Download the GitHub desktop installer that matches your operating system.
- Run the installer on your computer once it has finished downloading.
- Open GitHub desktop on your computer once it has finished installing.

Step 3 – Access Jupyter Open

To use Jupyter Open, you only need your Campus-wide login.

- Google "UBC Jupyter Open"
- Click the link that says "UBC JupyterHub Instructor Guide"
- Scroll down and you should find a button that says 'Log in to Jupyter Open'
- Enter your CWL information, and you should be all set to start coding.

Quick Aside: Jupyter Open

I am not sure how long your CWL's will be active after VSP ends.

- To continue using Python and Jupyter after they are deactivated, you will need to download them or find another cloud-based service.
- We can go over local installation near the end of class if there is interest, but there are many guides online that cover installation.
- Almost everything you learn on Jupyter Open will be exactly the same when using a local installation of Jupyter notebook.
- Jupyter Open includes file storage and even interfaces with GitHub but since you probably will not have access to it forever, we're going to use it exclusively as an IDE.

Step 4 – Create Jupyter Notebook

Jupyter Open should automatically generate a blank notebook for you

- Click on one of the cells and type " $x = 3$ " into the first cell without the quotes
- Go to the file drop down menu and click "Save notebook as". Save the file as "test_notebook.ipynb"
- Find the file on your file navigator on the left-hand side pane, right click it, and click "download" from the drop down menu.

Step 5 – Create Your Own Course Repository

You are now going to create your own repository where you will store your lab assignments, final project, and any other course materials you might want.

- Create a folder for this class on your computer
- Now open GitHub desktop and click on the file drop down menu and select "New repository"
- Title the repository "VSP_your_name"
- For local path, select the folder you created for this class
- After this, press create repository. Then push "publish this repository."
- Log on to GitHub via a browser to see your repository

Step 6 – Commit and Push a Change to Repository

Now that the repo is created, we will learn how to push changes we make in the local repository to the cloud.

- First, we have to make a change. Move "test_notebook.ipynb" into the directory "VSP_your_name"
- Open GitHub Desktop and navigate to the repository using the drop down menu on the left-hand side. You should see the changes you've made.
- Click commit – this prepares your files to be pushed to the cloud.
- Click push to finalize these changes and push them to the cloud.
- Log on to GitHub via a browser to see the changes.

Step 7 – Clone Course Repository

Now we are going to clone the course repository

- First, navigate to the course GitHub page. You can do this by searching my username "Joshua Catalano JayCata" on Google.
- Navigate to my repositories and click on "Working-With-Data-VSP-2023."
- Copy the URL of this repository
- Open GitHub Desktop, click on the file drop down, and click "clone repository"
- Paste the URL in the first box and set your local path to your class folder (the same folder your repository is in)